Solutions

Answer 8.1

Entry 0:

Entry 1: [10, 19, 28]

Entry 2: [20]

Entry 3: [12]

Entry 4:

Entry 5: [5]

Entry 6: [33, 15]

Entry 7:

Entry 8: [17]

Answer 8.1 The answer to each question is given below.
1. In the first pass (Pass 0), $\lceil N/B \rceil$ runs of B pages each are produced, where N is the number of file pages and B is the number of available buffer pages:
(a) $\lceil 10000/3 \rceil = 3334$ sorted runs.
(b) $\lceil 20000/5 \rceil = 4000$ sorted runs.
(c) $\lceil 2000000/17 \rceil = 117648$ sorted runs.

2. The number of passes required to sort the file completely, including the initial sorting pass, is $\lceil \log_{B-1} N1 \rceil + 1$, where $N1 = \lceil N/B \rceil$ is the number of runs produced by Pass 0:

(a) $\lceil \log_2 3334 \rceil + 1 = 13$ passes.
(b) $\lceil \log_4 4000 \rceil + 1 = 7$ passes.
(c) $\lceil \log_{16} 117648 \rceil + 1 = 6$ passes.

3. Since each page is read and written once per pass, the total number of page I/Os for sorting the file is $2 * N * (\#passes)$:

(a) 2*10000*13 = 260000.
(b) 2*20000*7 = 280000.
(c) 2*2000000*6 = 24000000.

4. In Pass 0, $\lceil N/B \rceil$ runs are produced. In Pass 1, we must be able to merge this many runs; i.e., $B - 1 \geq \lceil N/B \rceil$. This implies that B must at least be large enough to satisfy $B * (B - 1) \geq N$; this can be used to guess at B, and the guess must be validated by checking the first inequality. Thus:

(a) With 10000 pages in the file, B = 101 satisfies both inequalities, B = 100 does not, so we need 101 buffer pages.
(b) With 20000 pages in the file, B = 142 satisfies both inequalities, B = 141 does not, so we need 142 buffer pages.
(c) With 2000000 pages in the file, B = 1415 satisfies both inequalities, B = 1414 does not, so we need 1415 buffer pages.

Answer 8.3 The answer to each question is given below.

1. Assuming that the general external merge-sort algorithm is used, and that the available space for storing records in each page is 512−12 = 500 bytes, each page can store up to 10 records of 48 bytes each. So, 450 pages are needed in order to store all 4500 records, assuming that a record is not allowed to span more than one page. Given that 4 buffer pages are available, there will be $\lceil 450/4 \rceil = 113$ sorted runs (sub-files) of 4 pages each, except the last run, which is only 2 pages long.

2. The total number of passes will be equal to $\log_3 113 + 1 = 6$ passes.

3. The total I/O cost for sorting this file is $2 * 450 * 6 = 5400$ I/Os.

4. As we saw in the previous exercise, in Pass 0, $\lceil N/B \rceil$ runs are produced. In Pass 1, we must be able to merge this many runs; i.e., $B - 1 \geq \lceil N/B \rceil$. When B is given to be 4, we get N = 12 pages. The maximum number of records on 12 pages is $12 * 10 = 120$. When B = 257, we get N = 65792, and the number of records is $65792 * 10 = 657920$.

5.

    a. If the index uses Alternative (1) for data entries, and it is clustered, the cost will be equal to the cost of traversing the tree from the root to the leftmost leaf plus the cost of retrieving the pages in the sequence set. Assuming 67% occupancy, the number of leaf pages in the tree (the sequence set) is $\lceil 450/0.67 \rceil = 672$ (plus the cost of traversing the tree from the root to the leftmost leaf which is relatively minor comparing to this and thus we ignore it here for the sake of simplicity).

    b. If the index uses Alternative (2), and is not clustered, in the worst case, we scan B+ tree's leaf pages and for each data entry we also fetch a data page. The number of data entries is equal to the number of data records, which is 4500. Since there is one data entry per record, each data entry requires 12 bytes (4 bytes for the key and 8 bytes for the rid), and each page holds 512 bytes, thus the number of B+ tree leaf pages is about $(4500 * 12)/(512 * 0.67)$, assuming 67% occupancy, which is approximately 158. Thus, about 4658 I/Os are required in the worst-case scenario. (Plus the relatively minor cost of finding the leftmost leaf to start the scan, which we ignore here)

    c. The B+ tree in this case has 65792/0.67 = 98197 leaf pages if Alternative (1) is used and assuming 67% occupancy. This is the number of I/Os required

(ignoring the relatively minor cost of going from the root to the left-most leaf). If Alternative (2) is used, and the index is not clustered, the number of I/Os is approximately equal to the number of data entries in the worst case, that is 657920, plus the number of B+ tree leaf pages 23015. Thus, the number of I/Os is approximately 680935 (plus the relatively minor cost of finding the leftmost leaf, which we ignore here).